Arguments in favour of admission control for TCP flows

L. Massoulié and J. Roberts
CNET-France Télécom
38-40 rue du Général Leclerc, 92794 Issy-Moulineaux Cédex 9, France
Tel: (33) 1 45 29 57 01, Fax: (33) 1 45 29 65 56.
{laurent.massoulie, james.roberts}@cnet.francetelecom.fr}

Abstract

We advocate the use of admission control to limit the number of TCP flows on a network link to ensure that each has a minimal acceptable throughput. We demonstrate that, in the absence of such a control, the ineffective traffic due to the retransmission of lost packets constitutes a significant overhead and can even lead to congestion collapse in certain configurations. A second cause of inefficiency is the incomplete transmission of documents whose transfer is abandoned by users or higher protocol layers experiencing very low throughput. Admission control removes the cause of flow interruption, maintaining goodput even in case of demand overload. Finally, we discuss implementation issues, recognizing that the proposed admission control procedure is at the limit of current technology.

1 Introduction

Traffic in a multiservice network can be broadly classified as "stream" or "elastic": stream flows result from audio and video applications and require the network to preserve time integrity; elastic flows are established for the transfer of digital documents (files, pictures, ...) and only have loose response time requirements [Rob98a].

Admission control consists in refusing a new flow if the addition of its traffic would lead to an unacceptable quality of service level for that or any previously accepted flow. This type of control is frequently considered to be necessary for stream traffic, precisely in order to preserve time integrity, but not so for elastic traffic. Indeed, in defining the notion of elastic traffic, Shenker assumes that users derive a certain utility from an elastic flow which is a positive, strictly concave function of allocated bandwidth. This implies that overall utility gained from elastic traffic handled by a network link increases with the number of flows [She95]. In this paper we argue to the contrary that admission control is essential to preserving the efficiency of the network and the quality of service offered to users. In the absence of admission control, overload and consequent instabilities can lead to severe forms of congestion collapse. There *is* a minimum acceptable throughput below which users gain no positive utility.

We place our discussion in the context of the Internet where the large majority of traffic is elastic and is transferred under the end-to-end control of TCP. One function of TCP is to share bandwidth fairly among currently active flows. The mechanism by which this is achieved relies on individual flows gradually increasing their sending rate until a packet loss is detected, then decreasing the rate immediately to a lower level before starting a new gradual increase phase. The packet loss rate increases as the bandwidth share decreases. Since lost packets are retransmitted, there is a disproportionate increase in ineffective traffic as the number of flows sharing a bottleneck link increases. The ineffective traffic due to retransmissions can, in certain network configurations, lead to instability and eventual congestion collapse.

Congestion collapse occurs more obviously when traffic offered to a bottleneck link is greater than its capacity. By traffic demand, we mean the flow arrival rate multiplied by the expected volume of data to be transferred, assuming (for the sake of simplicity) that the flow arrival rate is not influenced by network status. If users do acquire utility from any positive bandwidth, however small, the number of flows in progress would increase indefinitely as long as the overload lasts: flows last longer and longer as their bandwidth gets smaller while the arrivals of new flows continues unabated. In practice, of course, as transfer times grow longer some users will grow impatient. Similarly, as bandwidth tends to zero and packet loss increases, some applications will assume the TCP connection is broken and abandon. In either case, all the work done in partially transferring a document is wasted and further contributes to congestion as users will often repeat their document request later.

The use of admission control to ensure any accepted flow receives a minimum acceptable throughput avoids wasting network resources on retransmissions and incomplete transfers and maintains useful throughput even in case of demand overload. In addition, we note that the mechanisms necessary to perform admission control are also required to perform intelligent routing of elastic flows, e.g., choosing the network path which offers the best throughput to a newly arriving flow [OO98]. It may also be argued that admission control is necessary in a network employing usage based charging [Rob98b]: if users pay in relation to the number of packets transmitted, the number of retransmissions should be negligible and any document transfer which begins should be guaranteed completion within a reasonable response time.

While we believe the above constitute compelling arguments in favour of introducing admission control, it remains to demonstrate that this is feasible. The majority of elastic flows are very short and the introduction of a classical signalling exchange to perform admission control would appear to constitute an unacceptable overhead. For TCP flows in the Internet, we suggest that an on-the-fly decision to accept or discard the first packet of a flow would be sufficient. It is then necessary, however, to be aware of the identities of currently active flows and to be able to classify packets according to these identities as and when they arrive. The necessary operations, for the envisaged high speed links of the future Internet, appear to be at (or just beyond) the limits of present technology [KLS98].

In the next section we discuss the nature of elastic traffic as a stochastic process and introduce a model of the TCP congestion avoidance mechanisms. In Section 3 we present simple mathematical models which illustrate the inefficiency and potential instability occasioned by the use in TCP of repeated packet loss as the means of determining available bandwidth. Section 4 then demonstrates how the interruption of transfers, due to user impatience or a broken higher layer connection, removes instability in case of overload but leads to considerable bandwidth wastage. Finally, in Section 5, we discuss the implementation issue, notably identifying further possible advantages of performing per flow queueing on the (limited) number of admitted flows.

2 TCP flow traffic model

The arguments for introducing TCP flow admission control are essentially derived from the observation that the network otherwise behaves in an inefficient and potentially unstable manner. To illustrate the inefficiencies and possible causes of congestion collapse we need a traffic model representing the random fluctuations in the number of flows and accounting for the way TCP shares link bandwidth between these flows. This model is necessarily a compromise between a realistic representation and mathematical tractability. In this section we discuss known traffic characteristics and TCP behaviour and introduce a number of simplifying assumptions.

2.1 Traffic characteristics

Despite the extreme growth and changing nature of the Internet, it is possible to derive from the results of a number of extensive measurement campaigns, a fairly general qualitative traffic characterization at the flow level [FGWK98]. TCP flows are in large majority generated in Web sessions whose starting times in a one hour time frame, say, can be accurately represented by a Poisson process. Within a session, a user retrieves a certain number of pages, each page possibly requiring the establishment of a number of TCP flows. The number of flows initiated in a session is highly variable (infinite variance distribution) producing self similarity in the flow arrival process, at least at the LAN/Internet interface [FGW98].

For the sake of simplicity, we will nevertheless assume the flow arrival process at a considered bottleneck link is Poisson. This is not necessarily incompatible with the experimental results, bearing in mind that the observed TCP flows do not all concern servers accessed via the considered link. A Poisson process results naturally when a very large population of users independently make relatively widely spaced demands. Moreover, the essential characteristic of the arrival process for our discussion is that its intensity has an exogenous component, whether it be the arrival rate of sessions or pages, which is independent of the network status.

The volume of data transferred in a TCP flow is highly variable. Measurements performed on Web pages [AW96, BC96] reveal a heavy tailed distribution with infinite variance: $\Pr[size > x] \sim x^{-\beta}$ for large x, where $1 < \beta \leq 2$. While some of our results are insensitive to the exact nature of this distribution, for reasons of tractability, we frequently have to make the assumption that the size distribution is exponential.

2.2 Modelling TCP

TCP is a complex window based protocol ensuring reliable data delivery by retransmitting lost (and errorred) packets. Its congestion avoidance algorithm aims to minimize loss and to realize fair sharing by dynamically adjusting the window size in response to network congestion. A fairly comprehensive model of TCP is described by Padhye et al. [PFTK98]. In particular, they derive an expression relating the flow throughput to the packet loss rate p. The flow rate $B(p)$ out of a TCP source in packets per second, including retransmissions, is given approximately by:

$$B(p) \approx \min\left(\frac{W_{\max}}{RTT}, \frac{1}{RTT\sqrt{2p/3} + T_0 \min\{1, 3\sqrt{3p/8}\} p(1+32p^2)}\right) \qquad (1).$$

where $W_{\max}$ is the receive window size, RTT is the round trip time and T_0 the initial time-out value.

Note that as $p \longrightarrow 1$, the flow rate does not tend to zero but to a certain limit value. In extreme congestion, TCP emits packets one at a time, on the expiration of a time-out which increases to the maximum value of $64T_0$. The approximation (1) is accurate for smaller values of p but overestimates the minimum throughput as $p \longrightarrow 1$.

When several flows share a given link, the throughput attained by each flow depends on many factors including the individual round trip times and the rate available on other links on the flow paths. Bandwidth sharing in a network is a complex subject [MR98a, RM98]. For present purposes, we will consider an isolated bottleneck link and assume that the mechanisms of TCP ensure that each flow achieves an exact fair share, i.e., if n flows are in progress on a link of capacity C, then each flow is transferring data at rate C/n.

3 Impact of retransmissions

In this section we consider the impact of retransmissions on congestion. In the case of an isolated bottleneck link, retransmissions have no an impact on the stability of that link but do add extra load to upstream network elements. However, we demonstrate that in more general network configurations with a particular traffic coupling between two or more links, congestion collapse can occur on those links simply because of the retransmissions.

3.1 Single bottleneck link

Consider a link of capacity C (bits/sec) handling TCP flows arriving according to a Poisson process of intensity λ (flows/sec) and transferring documents of mean size $1/\mu$ (bits). Let ρ denote the link load, $\rho = \lambda/C\mu$. By the assumption of perfect fair sharing, the number of flows in progress behaves like the customer population in a processor sharing system and is therefore geometrically distributed: $\Pr[n \text{ flows}] = \rho^n(1-\rho)$ if $\rho < 1$. This result is insensitive to the distribution of document sizes.

In case of overload ($\rho \geq 1$) this system is clearly unstable and congestion collapse is inevitable. Suppose here that we have $\rho < 1$ and consider the volume of retransmissions when the number of active flows is n. Let γ' be the input rate in bits/sec into the bottleneck link of one flow. The loss rate is then p such that $\gamma' = B(p)L$, where L is the packet size in bits and $B(p)$ is given by (1). By conservation, we also have the relation $C/n = \gamma'(1-p)$. Solving the two simultaneous equations allows us to calculate the loss rate p and consequently, the input rate γ' as a function of n. Figure 1 plots the relative retransmission overhead $(n\gamma'/C - 1)$ as a function of the current flow transfer rate C/Ln, expressed here in packets per second. These results are independent of the particular link rate C and packet size L and are derived using the parameter values $\{RTT = 0.257, T_0 = 1.454, W_{\max} = 33\}$ corresponding to one connection ("pif to manic") reported in [PFTK98]. They demonstrate that the overhead, representing the extra work which must be accomplished by the upstream network elements (including the router processor giving access to the considered link), grows rapidly as the flow transfer rate decreases. Employing admission control to ensure a transfer rate of at least 6 packets per second (around 24 Kbit/s) would limit the overhead to 10%.

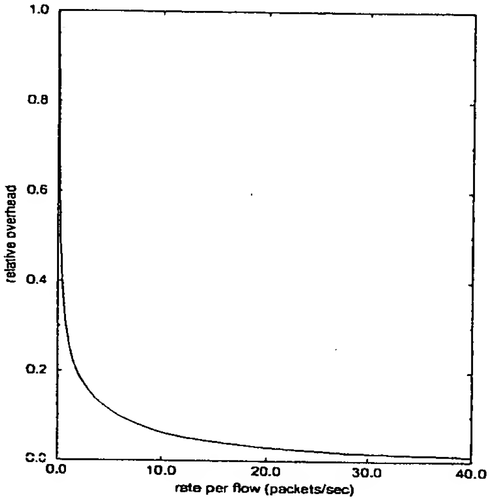

Figure 1: Retransmission overhead as a function of flow rate (packets/sec)

3.2 Traffic coupled links

In certain network configurations, the retransmission of lost packets can lead to congestion collapse even when the offered load is less than capacity. This can happen, for example, when flows on one traffic route use two links in one order while flows on another route use the same links in the inverse order. This example is indicative of a more general configuration where links are joined in a closed chain by a set of routes using two or more adjacent links, as depicted in Figure 2. We consider the two-link ring, first assuming a simple traffic model to illustrate the instability phenomenon before indicating how congestion collapse can occur with TCP flows.

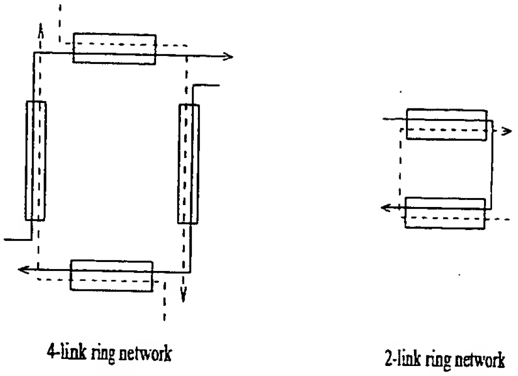

Figure 2: Ring networks with feedback loop

3.2.1 A preliminary model

Consider the two-link ring network of Figure 2. There are two servers 1 and 2, and two traffic routes 1 and 2, traffic route i, $i = 1, 2$, first visiting server i and then the other server. Assume that traffic on each route arrives in a fluid manner at constant rate γ', that service at each server is FIFO at normalized speed 1, and that buffers are finite. Rate γ' is meant to represent the total arrival rate at the first server on the route due to a possibly large number of individual flows, and includes retransmissions. The effective throughput on each flow is denoted γ. Let p be the proportion of fluid lost at each server. Because of the FIFO assumption, the rate of traffic entering server 1 (resp. 2) along route 2 (resp. 1) is $\gamma'(1-p)$; thus p satisfies

$$1 - p = \begin{cases} 1 & \text{if } \gamma' \leq 1/2, \\ 1/(\gamma' + \gamma'(1-p)) & \text{if } \gamma' > 1/2, \end{cases}$$

yielding

$$1 - p(\gamma') = \begin{cases} 1 & \text{if } \gamma' \leq 1/2, \\ \frac{1}{2}(-1 + \sqrt{1 + 4/\gamma'}) & \text{if } \gamma' > 1/2. \end{cases}$$

We also have the flow conservation relation $\gamma = (1 - p(\gamma'))^2\gamma'$ which, together with the above, defines a function $\gamma(\gamma')$ which is plotted in Figure 3.2.1.

Figure 3: Throughput γ as a function of input rate γ'.

Assume now that the throughput is a pre-specified value $\gamma \in [0, 1/2)$. It is readily seen on Figure 3.2.1 that there exist two distinct values for γ' corresponding to the given value γ, the lowest one, γ'_-, being exactly γ, and the largest one, γ'_+, corresponding to a non-zero loss regime ($\gamma'_+ = (1-\gamma)^2/\gamma$).

In the considered model, for a given demand γ, the input rate is constant and equal to either γ or γ'_+. Assume now that there is a constant delay δ between loss and retransmission. The input rate at time t. $\gamma'(t)$ then evolves according to the equation:

$$\gamma'(t) = \gamma + p(\gamma'(t-\delta))\gamma'(t-\delta) + [1 - p(\gamma'(t-\delta))]\,p(\gamma'(t-\delta))\,\gamma'(t-\delta) \tag{2}$$

An elementary stability analysis allows us to conclude that the equilibrium point $\gamma' = \gamma$ is stable, while $\gamma' = \gamma'_+$ is not: a small perturbation to the left leads the trajectory to γ, while a small perturbation to the right results in an explosive trajectory. We thus conjecture that, in the presence of stochastic perturbations, the system will linger for a long time in the vicinity of the attractive point $\gamma' = \gamma$, which then corresponds to a metastable state, but eventually sufficiently strong perturbations will result in unstable behaviour. This observation is pursued in the next subsection.

3.2.2 Instability for sources implementing TCP's congestion avoidance algorithm

Consider again the two-link ring network of Figure 2, and assume now that new flows on route i arrive at rate λ_i, $i = 1, 2$, and last for the transfer of a volume of data which is exponentially distributed with parameter μ_i. Assume that Equation (1) of Padhye et al. [PFTK98] applies to this situation, so that, if p_i is the loss proportion on link i, the rate (in packets/sec) at which type i connections emit data is exactly

$$\begin{aligned} v_1 &= B\,(p_1 + (1-p_1)p_2)\,, \\ v_2 &= B\,(p_2 + (1-p_2)p_1)\,. \end{aligned} \tag{3}$$

By conservation, the quantities v_i and p_i also satisfy

$$\begin{aligned} (1-p_1)(n_1v_1 + (1-p_2)n_2v_2) &= 1, \\ (1-p_2)(n_2v_2 + (1-p_1)n_1v_1) &= 1. \end{aligned} \tag{4}$$

The four previous equations determine these quantities in terms of n_1 and n_2. The process (X_1, X_2) counting the number of ongoing connections along each route is then a Markov process, with transition rates given by

$$\begin{aligned} n_i &\to n_i + 1 : \ \lambda_i, \\ n_i &\to n_i - 1 : \ \mu_i v_i n_i (1-p_1)(1-p_2). \end{aligned}$$

We then have the following result.

Proposition 1 . *The Markov process (X_1, X_2) is transient, provided both λ_1 and λ_2 are positive, and the function B is lower bounded by some $v_{min} > 0$.*

The proof relies on Lyapunov function techniques (more precisely, on Foster's transience criterion, as described in Asmussen [Asm87]), and can be found in [MR98b]. This instability property can be explained as follows: if n_1, say, is large, then because the sending rate of type 1 connections is lower bounded by v_{min}, the arrival rate to server 1 is larger than $n_1 v_{min}$, and thus p_1 is close to 1. Type 2 connections, which feed server 1 at a rate not larger than 1 (the output rate of server 2), then have a goodput not larger than $(1-p_1)$, which is close to zero. Hence the number of type 2 connections builds up, resulting in a value of p_2 close to 1, so that type 1 connections are blocked at server 2, and so on.

3.3 Discussion

First notice that the results derived for two coupled links also apply to a symmetric chain of an arbitrary number of links. The same developments can obviously be extended to more general asymmetric configurations and they would apply also if an arbitrary number of non-bottleneck links were included. In other words, although the considered model is somewhat contrived in the interests of simplicity, it is representative of real life network configurations. We conjecture, therefore, that this instability phenomenon is indeed present in the current Internet. Instability due simply to overload, as discussed in the case of an isolated link, also clearly exists in the network.

While the instability phenomenon is real, a number of factors mitigate against its effects being visible. Firstly, the network does not generally perform the detailed traffic measurements necessary to identify retransmissions or allow an estimation of the number of flows currently active. Special measurement campaigns do reveal very large numbers of flows on some links [TMW97] or excessively high loss rates [PFTK98]. The feedback instability, while theoretically possible, will only occur rarely at low loads. An analogy with the well known instability phenomenon of Aloha is appropriate, although the assumption of Poisson arrivals is more reasonable here than in the finite source environment in which Aloha is usually employed. Finally, instability is naturally limited by the behaviour of users or end applications which are not infinitely patient. The additional impact on network efficiency of this behaviour is discussed below.

4 Incomplete transfers

When rate tends to zero, either customers are impatient (they click on the browser stop button) or applications interpret the very high loss rate and long inter-packet times as evidence of a broken connection and abandon the transfer. An instance of impatience is illustrated in Figure 6 in [FGWK] where a group of 6 TCP flows (presumably corresponding to different elements of a single URL) are interrupted simultaneously after 30 minutes. Note, however, that the stop button does not necessarily interrupt a transfer on a network link when the transfer from a Web server takes place via a proxy: even though a user grows impatient, the proxy will continue to download the requested document and store it locally for future consultation. In the latter case, transfers are only interrupted if the application implemented by the proxy server decides that throughput is so low and the loss rate so high that the connection must have broken.

4.1 User impatience

We reconsider the processor sharing model introduced above where, however, we introduce user impatience. Specifically, we now assume document size and user patience are both exponentially distributed. These assumptions are clearly not realistic but seem necessary to derive a tractable model.

4.1.1 Markovian model

Assume then that a single server with unit service speed (i.e., $C = 1$) attends customers in a PS fashion. Customers arrive to the queue according to a Poisson process with rate λ. Their service requirements constitute an i.i.d. sequence of exponentially distributed random variables with mean $1/\mu$. Customers may grow impatient and leave the queue before their service is complete. The sequence of the "patience durations" of the customers is an i.i.d. sequence of exponential random variables with mean $1/\mu_i$; this sequence is independent of the other sources of randomness. We assume admission control is implemented by rejecting additional customers when there are already N customers present in the system.

Denote by X_t the number of customers in the system at time t. Clearly, $\{X_t\}$ is a Markov process on $\{0, \ldots, N\}$, with transition rates given by

$$\begin{cases} q(n, n+1) = \lambda, \ 0 \leq n < N; \\ q(n, n-1) = \mu + n\mu_i, \ 0 < n \leq N. \end{cases}$$

It has the stationary probability distribution $\{\pi_n\}$ defined by

$$\pi_n = \pi_0 \left(\frac{\lambda}{\mu_i}\right)^n \frac{1}{\prod_{k=1}^{n}(k + \mu/\mu_i)} \tag{5}$$

where

$$\pi_0 = \left(\sum_{n=0}^{N} \left(\frac{\lambda}{\mu_i}\right)^n \frac{1}{\prod_{k=1}^{n}(k + \mu/\mu_i)}\right)^{-1} \tag{6}$$

The state variable X_t is not enough to measure the efficiency of the server. Consider the quantities θ_n, $n = 1, \ldots, N$, defined as the mean work spent by the server on the customers present in the system, given that X_t is currently jumping from state n to another state (we are interested in the work done on the customers present *before* the jump). Consider the imbedded discrete time Markov chain Y_n, which consists of the sequence of states visited by X_t. It is reversible, with stationary measure $\tilde{\pi}_n$ satisfying

$$\tilde{\pi}_n \propto (\mathbf{1}_{n<N}\lambda + \mathbf{1}_{n>0}(n\mu_i + \mu))\pi_n, \ 0 \leq n \leq N$$

Its non-zero transition probabilities are

$$\begin{cases} d(n, n+1) = \lambda/(\lambda + \mathbf{1}_{n>0}(n\mu_i + \mu)), \ 0 \leq n < N; \\ d(n, n-1) = (n\mu_i + \mu)/(\mathbf{1}_{n<N}\lambda + n\mu_i + \mu), \ 0 < n \leq N. \end{cases}$$

The quantities θ_n, $n = 1, \ldots, N$ satisfy the system of linear equations:

$$\theta_n = \frac{1}{\mathbf{1}_{n<N}\lambda + \mathbf{1}_{n>0}(n\mu_i + \mu)} + d(n, n+1)\frac{n}{n+1}\theta_{n+1} + d(n, n-1)\theta_{n-1} \tag{7}$$

where $\theta_0 = \theta_{N+1} = 0$. To see this, consider an instant T^- at which X_t is about to leave state n. The work done on the customers at that time is the sum of the time spent in state n and the work done on the customers prior to the entrance into state n. The first term in the above expression of θ_n is exactly the mean sojourn time in state n. By reversibility of the chain $\{Y_k\}$, knowing that the system is in state n, the probability that the previously visited state was $n \pm 1$ is exactly $d(n, n \pm 1)$.

In case the former state is $n-1$, no customer departure occurred at the previous transition and the average work accomplished before the jump from $n-1$ to n, on the customers still in the system at T^-, is then θ_{n-1}. In case the former state is $n+1$, a customer departure occurred, and the choice of the customer departing was completely random, not depending on the work already received by this customer, so that the average work achieved before the jump from $n+1$ to n on the customers still in the system at T^- is given by $\theta_{n+1}n/(n+1)$.

We now turn to the computation of server efficiency. The arguments are only sketched here, and a more careful derivation can be found in [MR98b]. The useful work done by the server per time unit, which we denote by U, is given by

$$U = \frac{1}{t}\mathbf{E}\,(\text{work done on customers completing service in } [0,t])$$

for all $t > 0$. Conditionally on $X_0 = n$, the probability of having one transition during $[0,t]$ is given by $(\lambda\mathbf{1}_{n<N} + n\mu_i + \mu\mathbf{1}_{n>0})t + o(t)$. Conditionally on both events, the amount of work done on these n customers prior to the transition is, on average, θ_n. Also, conditionally on these two events there is a probability

$$\mu\mathbf{1}_{n>0}/(\lambda\mathbf{1}_{n<N} + n\mu_i + \mu\mathbf{1}_{n>0})$$

that this transition corresponds to a service completion, in which case the average amount of useful work done on the departing customer is θ_n/n. Summarizing, one obtains

$$U = \frac{\sum_{n=1}^{N}\pi_n(\lambda\mathbf{1}_{n<N} + n\mu_i + \mu\mathbf{1}_{n>0})t\,[\mu/(\lambda\mathbf{1}_{n<N} + n\mu_i + \mu\mathbf{1}_{n>0})][\theta_n/n] + o(t)}{t}$$

Letting $t \to 0$ yields

$$U = \mu\sum_{n=1}^{N}\pi_n\theta_n/n \tag{8}$$

Formula (8) can be used to investigate numerically how the efficiency of the system under consideration varies with N.

4.1.2 Numerical application:

Displayed in Figure 4 is the behaviour of U as a function of N, for three offered loads, unit sized documents ($\mu = 1$) and an assumed impatience rate of $\mu_i = 0.01$. The figure clearly illustrates the existence of a maximum goodput obtained by limiting the number of simultaneous flows. The precise value of the maximum depends on the chosen parameters (μ_i, in particular) and is not particularly significant for present purposes. In Figure 5, we show how goodput depends on the impatience rate. In the practically significant case where the impatience rate is small, it may be noted that reasonable goodput is obtained for a wide range of admission control thresholds, suggesting that the value of N is not highly critical. All curves show that goodput tends to a limit as N increases. For patient users (μ_i small) or heavy traffic ($\lambda >> \mu$), the value of this limit is $1/\rho$ (this is formally established in [MR98b]) which, as confirmed in the figures, constitutes a reasonable approximation for other relevant parameter ranges.

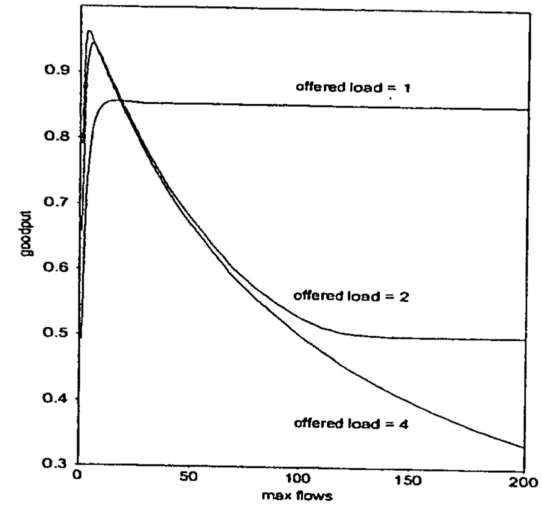

Figure 4: Goodput against admission threshold for impatience rate μ_i =0.01

4.2 Broken connections

An alternative reason for flows not completing is that the higher layer applications deduce from the poor performance of TCP (low throughput, excessive loss) that the connection is broken and consequently interrupt the transfer. A simple approach to evaluating this behaviour is to adopt the above processor sharing model and to assume that, whenever the number of flows in progress exceeds a limit value N, one of the flows immediately breaks, the instantaneous throughput $1/(N+1)$ being insufficient to sustain the application layer protocol (we do not assume all connections are then broken but rather that the first one to break immediately releases the pressure on the others).

The analysis of the preceding section still applies on setting $\mu_i = 0$ everywhere, and replacing the condition $\theta_{N+1} = 0$ by the equation $\theta_{N+1} = \theta_N$. The latter equation reflects the fact that the number of flows attains $N+1$ only for an infinitesimal duration (the time for one connection to break) before reverting to the value N. With these modifications we can again evaluate the useful utilization U by (8).

Numerical results show that for any reasonably large value of N, we have $U = \rho$ for $\rho < 1$ and $U = 1/\rho$ for $\rho > 1$. Simply setting an admission control threshold large enough but less than N is sufficient to maintain the value of U at 1 in case of overload.

4.3 Discussion

The above models are clearly very simple and imprecise. However, we maintain that they do amply illustrate the following two points:

- useful work accomplished by a bottleneck link in case of overload ($\rho > 1$) can be much less than the observed utilization (as low as $1/\rho$ with the above Markovian assumptions);

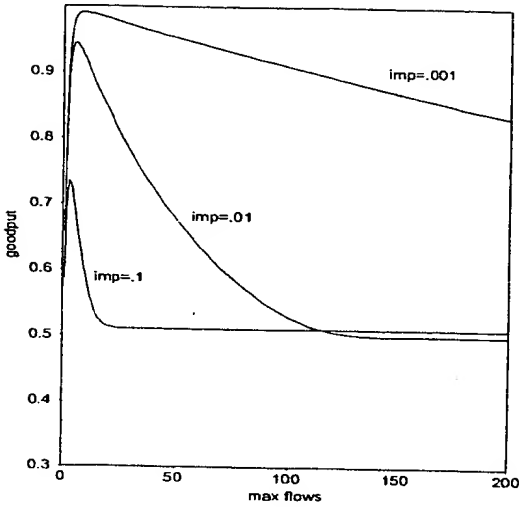

Figure 5: Goodput against admission threshold for offered load = 2 (μ_i = imp).

- admission control, by limiting the number of flows in progress to be below a suitably chosen threshold, is an effective means to maintain useful throughput.

5 Realizing admission control

While we believe the arguments of the previous sections clearly show the desirability of admission control for TCP flows, it remains necessary to demonstrate that this is feasible. In this section we describe the type of admission control we have in mind and discuss the realization issue.

5.1 Counting flows

Given the very small size of most documents transferred using TCP, it is clearly unreasonable to implement admission control using signalling exchanges which introduce significant overhead and latency. Rather, we envisage a lightweight procedure where admission control and flow routing are performed "on the fly". A router would need to be able to identify whether an incoming packet belongs to an existing or a new flow. In the former case, the packet would be forwarded to a predefined output link (as in MPLS, for example [VFZC98]). Admission control and flow routing are performed for new flows. The router would need to determine an appropriate outgoing interface, taking account of the required destination and implementing some to-be-defined routing policy. If no such egress is suitable, the router would simply not forward the packet. With TCP, the latter action is equivalent to admission refusal. Re-attempts will be made and any subsequent rejections interpreted according to the protocol specification.

Flow routing is beyond the scope of the present paper and we limit subsequent discussion to the issues of identifying a new flow and deciding if a given outgoing link can accept it. Assume the packets of a given TCP flow all enter the router on a particular input. To determine whether a flow is new or not, it is sufficient to compare its identifier (5-tuple) with that of the flows on a list of active flows. Flows would be removed from the list if no packet were received within a certain time-out interval, as envisaged in the case of IP-Switching [NLM96]. This is certainly not an easy operation but one which does not seem to be outside the realms of possibility, given recent technological developments [KS98, KLS98].

To avoid the instability and inefficiency problems considered above, the admission control decision can be based simply on knowledge of the number of flows currently active on an outgoing link. This number could be deduced from the history of previous routing decisions and assumed flow cessations on the incoming links.

5.2 Queue management

TCP has been designed to work with routers equipped with a simple FIFO queue on each outgoing link implementing tail drop packet discard. This is the assumed operating mode in the derivation of the throughput formula (1). Employing admission control with such routers would considerably improve performance, as discussed above. However, flow identification, which is necessary for admission control, opens the possibility for more sophisticated queueing schemes with potentially greater gains in efficiency, robustness and fairness.

We have in mind, for instance, the per flow fair queueing scheme described by Suter et al., [SLSC98]. As well as the advantages identified in that paper, there is evidence that per flow fair queueing at link level leads to overall max-min fair rate allocations in the network [MR98a]. Alternative scheduling policies sacrificing fairness in the interests of overall efficiency could also be envisaged.

5.3 TCP enhancement

Despite the fact that the success of the Internet owes much to the ubiquity and the robustness of the different versions of TCP developed over the last 10 years or so, this protocol still constitutes a rather inefficient means of sharing network bandwidth between concurrent elastic flows. The inefficiency of packet loss as an indicator of congestion is addressed in the current proposal to use an explicit congestion notification (ECN) indication [Flo94, RF98]. More radical enhancements, such as those proposed in TCP Vegas [BP95], can improve overall efficiency. However, they are unlikely to be introduced if, as at present, their use in competition with the more aggressive Reno and Tahoe versions would lead to less throughput for their users. Flow identification and per flow scheduling, by ensuring fair and efficient bandwidth sharing, can remove the current advantages gained by a implementing more aggressive reactions to congestion, leading to the development of a (yet) more friendly version of TCP.

6 Conclusions

We advocate the use of admission control for both stream (audio and video) and elastic (document transfer) flows. For elastic flows, admission control would be used to ensure that every accepted flow experiences a minimum acceptable throughput. While it may be considered that such quality of service is essential in any commercial network where users are required to pay for the document and/or its transport, the arguments developed here highlight the negative impact on network performance of not performing admission control.

We have first highlighted the considerable inefficiencies due to the use of packet loss as the means for TCP connections to probe for their share of network bandwidth. Retransmission of lost packets constitutes a considerable overhead (>10%) whenever flow throughput decreases below a rate of around 6 packets per second. To maintain such a minimal throughput would require admission control to limit the number of flows currently sharing any bottleneck link. In certain routing configurations, the network can enter an unstable state where almost all bandwidth is wasted handling retransmissions, while the number of flows in progress increases unboundedly. Such instability is checked by limiting the number of flows admitted. We note that the proposed use of explicit congestion notification would also remove this source of instability, as long as a sufficiently large proportion of flows were to use this enhancement to TCP.

A second source of inefficiency in a network without admission control is the propensity for flows to be interrupted as the response time is prolonged due to congestion. Since the work done transferring the first part of a document is generally wasted, incomplete transfers constitute a significant source of inefficiency. Simple models proposed in this paper illustrate that the proportion of link bandwidth wasted in this way can be considerable unless admission control is employed to maintain the throughput at an acceptable level.

The simple admission control condition of limiting the number of elastic flows in progress on a link to be below some threshold, nevertheless appears to be at the current limit of technological feasibility. We suggest that effort of developing an efficient solution would be well worthwhile, however, both for the identified advantages of admission control and for the enhanced possibilities afforded for intelligent flow routing. Fairness and efficiency could be further improved by the additional use of per flow queueing mechanisms.

References

[AW96] M.F. Arlitt and C. Williamson. Web server workload characterization: the search for invariants. ACM Sigmetrics 96, 1996.

[Asm87] S. Asmussen. *Applied Probability and Queues*. Wiley, 1987.

[BC96] A. Bestavros and M. Crovella. Self-similarity in World Wide Web traffic: evidence and possible causes. ACM Sigmetrics 96, 1996.

[OO98] S. Oueslati-Boulahia, E. Oubagha. An approach to routing elastic flows. Submitted, 1998.

[BP95] L.S. Brakmo and L.L. Peterson. TCP Vegas: end to end congestion avoidance in a global Internet. IEEE JSAC, Vol. 13, pp 1465-1480, 1995.

[FGWK98] A. Feldman, A. C. Gilbert, W. Willinger, T. G. Kurtz. The changing nature of network traffic: Scaling phenomena. Computer Communications Review, Vol 28, No 2, April 1998.

[FGW98] A. Feldman, A. C. Gilbert, W. Willinger. Data networks as cascades: Investigating the multifractal nature of Internet WAN traffic. SIGCOMM'98, 1998.

[Flo94] S. Floyd. TCP and Explicit Congestion Notification. ACM Computer Communication Review, V. 24 N. 5, October 1994, pp. 10-23.

[KS98] S. Keshav and R. Sharma. Issues and trends in router design. IEEE Commun. Mag., pp 144-151, May 1998.

[KLS98] V.P. Kumar, T.V. Lakshman and D. Stiliadis. Beyond best effort: router architectures for the differentiated services of tomorrow's Internet. IEEE Commun. Mag., pp 152-164, May 1998.

[MR98a] L. Massoulié and J. Roberts. Bandwidth sharing: objectives and algorithms. Submitted, 1998.

[MR98b] L. Massoulié and J. Roberts. Limits of the best effort traffic management paradigm. In preparation.

[NLM96] P. Newman, T. Lyon and G. Minshall. Flow labelled IP: a connectionless approach to ATM. INFOCOM'96, pp 1251-1260, 1996.

[PFTK98] J. Padhye, V. Firoiu, D. Towsley and J. Kurose. Modeling TCP throughput: a simple model and its empirical validation. ACM SIGCOMM'98.

[RF98] K.K. Ramakrishnan and S. Floyd. A Proposal to add Explicit Congestion Notification (ECN) to IP . Internet draft draft-kksjf-ecn-01.txt, July 1998, work in progress.

[Rob98a] J. Roberts. Realizing quality of service guarantees in multiservice networks. Proceedings of PMCCN'97, Chapman&Hall, 1998.

[Rob98b] J. Roberts. Quality of service guarantees and charging in multiservice networks. IEICE Trans. Commun., Vol E81-B, No 5, May 1998.

[RM98] J. Roberts and L. Massoulié. Bandwidth sharing and admission control for elastic traffic. ITC Specialist Seminar, Yokohama, October 1998.

[She95] S. Shenker. Fundamental design issues for the future Internet. IEEE JSAC, Vol 13, No 7, pp1176-1188, September 1995.

[SLSC98] B. Suter, T. V. Lakshman, D. Stiliadis, A. K. Choudhury. Design considerations for supporting TCP with per-flow queueing. Proceedings of INFOCOM'98, 1998.

[TMW97] K. Thompson, G.J. Miller and R. Wilder. Wide-area Internet traffic patterns and characteristics. IEEE Network., Nov/Dec 1997.

[VFZC98] A. Viswanathan, N. Feldman, Z. Wang and R. Callon. Evolution of multiprotocol label switching. IEEE Commun. Mag., pp 165-173, May 1998.